Call: NFRP-2018

(Nuclear Fission, Fusion and Radiation Protection Research)

Topic: NFRP-2018-11

Type of action: CSA

Project: “Fair4fusion – open access for fusion data in Europe”

D3.1 – Report on technology survey and demonstrator requirements

WP3

Deliverable status	Final
Type	Report
Dissemination level	Public
Work package	WP3 – Blueprint Architecture for Fusion Open Data
Lead beneficiary	NCSR-D
Due date	31 August 2020
Date of submission	31 August 2020

Project Name	Fair4fusion – open access for fusion data in Europe
Grant Agreement	847612
Project Duration	1 September 2019 – 31 August 2021

Document Information

AUTHOR

Author	Organisation	Contact
Antonis Koukourikos	NCSR-D	kukurik@iit.demokritos.gr
Andreas Ikonomopoulos	NCSR-D	anikon@ipta.demokritos.gr
Iraklis Angelos Klampanos	NCSR-D	iaklampanos@iit.demokritos.gr
Sissy Themeli	NCSR-D	sthemeli@iit.demokritos.gr
Vangelis Karkaletsis	NCSR-D	vangelis@iit.demokritos.gr
Bartosz Bosak	PSNC	bbosak@man.poznan.pl
Marcin Plociennik	PSNC	marcinp@man.poznan.pl
Raul Palma	PSNC	rpalma@man.poznan.pl
Frederic Imbeaux	CEA	Frederic.IMBEAUX@cea.fr
Shaun de Witt	UKAEA	shaun.de-witt@ukaea.uk

DOCUMENT CONTROL

Document version	Date	Author/reviewer - Organisation	Change
1	20/3/2020	Antonis Koukourikos – NCSR-D	Structure / ToC
2	27/3/2020	Antonis Koukourikos, Irakis Klampanos – NCSR-D	Section 2

3	3/4/2020	Antonis Koukourikos, Sissy Themeli – NCSR-D	Section 3
4	10/4/2020	Antonis Koukourikos – NCSR-D	Introduction, draft distribution for internal review
5	16/4/2020	Andreas Ikonomopoulos, Vangelis Karkaletsis – NCSR-D	First internal review
6	24/4/2020	Sissy Themeli, Antonis Koukourikos – NCSR-D	Updates based on first review
7	19/5/2020	Iraklis Klampanos, Andreas Ikonomopoulos – NCSR-D	Section 5
8	26/6/2020	Iraklis Klampanos – NCSR-D	Updates in Section 5
9	13/7/2020	Iraklis Klampanos, Vangelis Karkaletsis, Andreas Ikonomopoulos – NCSR-D	Finalisation of content and internal review
10	30/7/2020	Bartosz Bosak, Marcin Plociennik - PSNC	Updates across whole document
11	31/7/2020	Raul Palma – PSNC	Added annotations subsection
12	20/8/2020	Iraklis Klampanos and Andreas Ikonomopoulos – NCSR-D	Various edits across the document and addition of the requirements appendix
13	24/8/2020	Iraklis Klampanos and Andreas Ikonomopoulos – NCSR-D	Various edits
14	25/8/2020	Iraklis Klampanos and Andreas Ikonomopoulos – NCSR-D	Addressed comments of David Coster (MPG) as internal reviewer
15	31/8/2020	Iraklis Klampanos – NCSR-D	Addressed comments of Joan Decker (EPFL) as internal reviewer
16	31/8/2020	Iraklis Klampanos – NCSR-D	Addressed comments of Pär Strand (CTH) and of Marcin Plociennik (PSNC)

DOCUMENT DATA

Keywords	**Technology survey, demonstrators, requirements**
Point of Contact	Iraklis Klampanos (NCSR-D)
Internal reviewers	David Coster (MPG), Joan Decker (EPFL)
Delivery Date	31 August 2020

Table of Contents

Executive Summary

This deliverable reports on an initial survey and assessment of potentially suitable software and architectures to inform further consideration during the specification of the architectural blueprint. Further, it makes use of the findings of the survey as well as of the use cases and requirements drafted in WP2. It reiterates technical requirements which will be met by two demonstrators that focus on different technical challenges.

The technological survey covers authentication and authorization infrastructure (AAI), data representation and annotation, persistent storage, search and visualisation and front-end third-party and in-house (provided by Fair4Fusion or EUROfusion partners) technologies. A selection of these technologies is being used by the Fair4Fusion demonstrators with the intension to evaluate them based on functional and non-functional (e.g. expertise, extensibility, etc.) criteria, with the ultimate goal to inform the blueprint architecture (D3.2 : Blueprint Architecture for a Fusion Open Data Framework with proposition of the implementation plan).

This responds to the requirements elicited in WP2. These requirements are provided here as an Appendix. In WP5, the demonstrators, to the degree that is possible due to their nature and to external non-functional conditions (e.g. available resources, central decisions needed, etc.), will respond to at least these requirements designated by “MUST”.

1 Introduction

In the context of defining the overall architecture of a platform appropriate for rendering fusion data FAIR and as open as possible – the primary objective of WP3 – it is necessary to identify and assess the computational resources that could be integrated in the platform in order to fully cover the expectations of the targeted user groups. In conjunction with the fulfilment of the basic functional requirements, the adoption of basic FAIR principles and guidelines should be facilitated by the appropriate tool selection. From the technical point of view, this poses certain priorities in technology adoption, mainly in two directions:

i. Support for rich descriptions and associations between data items
ii. Provision of standardised access and sharing mechanisms

In accordance to these two assumptions that stem from the project strategic objective and the finalisation of the user needs as expressed in the context of WP2, T3.1 and T3.2 are to propose blueprints on which the core functionalities of the Fair4Fusion platform will be materialised.

Section 2 lays out the methodology followed for directing the technology survey, while Section 3 describes the main technologies that are suitable candidates for integration in the envisaged platform and makes a reference to the project technical requirements as those are listed in Appendix I. Section 4 juxtaposes the identified technologies with solutions already used by consortium partners and their communities, while Section 5 proceeds to formulate suggestions for technology adoption while highlighting selections requiring detailed testing before achieving technology selection status. The conclusions drawn from these works are summarised in Section 6.

2 Methodology

The driver for understanding the functionalities and, thus, the technical assets needed to be incorporated in the platform is the set of requirements produced in the context of T2.2, *Use Case Definitions for Open Science* and the corresponding deliverables, D2.2 *Interim Report on Open Science Use Cases for Fusion Information* and D2.3 *Final Report on Open Science use Cases for Fusion Information*. The analysis of the collected use cases / user stories – with respect to their technical requirements – showcased, in general, requirement overlapping (Appendix I) and grouping in specific technology classes/categories.

At the same time, specific to technology evaluation and the demonstration of potential with respect to data FAIRness for Fusion, there are non-functional requirements to be considered too. On one hand, EC progresses fast with the opening up of research data and methods in a multitude of fields, making use of e-infrastructures and their associated ecosystems, e.g. the European Open Science Cloud[1], etc. This is a fast-moving area in political as well as in technological terms. On the other hand, the EUROfusion Gateway (also referred to as the “Gateway”)[2] is a widely used shared resource specific to the Fusion community, which accumulates and builds on knowledge

[1] https://eosc-portal.eu
[2] Accessible via eufus.eu

and technology within its limits. It also serves as a vital communication vehicle amongst EUROfusion members in order to promote data and methodology schemas and standards. It is expected that a future solution that promotes FAIR and open science would need to take on board technologies and methodologies developed within the EUROfusion Gateway. Fair4Fusion members have extensive experience and involvement in both environments – the wider European/EOSC community and the EUROfusion Gateway – while they focus their efforts on using ITER Integrated Modelling & Analysis Suite (IMAS).

In order for the project outcomes to serve the community better in preparation for the next step, Fair4Fusion prepares two distinct demonstrators. The two demonstrators are aligned in terms of the main use-cases they address, while their user-interface is inspired by the JET dashboard. On the other hand, they focus on experimenting on different technical aspects also in line with the two distinct environments they primarily operate in.

Demonstrator 1 focuses on making as much use of the present Fusion technological ecosystem as possible, as this is available on the Gateway. This includes putting emphasis on:

- Data and metadata ingestion from various sites
- Using the demonstrator as a testbed for de-facto standards, such as the Summary IDS defined as part of the ITER Physics Data Model (PDM)
- Investigating the boundaries between the two systems, increasing the useful technological overlap

Demonstrator 2 focuses on exploring alternative technologies and approaches, especially to free text searching and to high-level metadata analysis for visualisation. This includes:

- Alternative metadata/Summary IDS representations
- Visualisation frameworks not explored within other user-facing systems, such as the JET dashboard
- The use of search technologies, e.g. Solr or ElasticSearch (see Sec. 3.4, below)

Both demonstrators are designed to explore and communicate different possibilities to the Community and inform future developments. They are implemented and documented as part of MS15 (met and submitted end of M12).

3 Requirements and Relevant Technologies

As a first step in the technology survey, the main technologies that ought to be foreseen in the Fair4Fusion architecture are summarised in Table 1. The technical requirements are provided as an appendix, with the demonstrators focusing on the “MUST” requirements.

Table 1: Technology requirements stemming from the Fair4Fusion Use Cases

Technology	**Description**	***FAIR* ref.**
Authentication and Authorization Infrastructure (AAI)	Services and underlying deployment mechanisms to ensure secure and consistent access to the infrastructure	A, I
Data representation and annotation	Standards and tools for semantically describing the relevant data	F, I, R
Persistent storage	The data storage mechanisms for the data itself, as well as their metadata descriptions. These metadata (summaries) are only used for discovery and high-level visualisation purposes. Persistent storage for experimental data remains the responsibility of the site.	F, I, R
Search and visualisation	Tools and frameworks for indexing data and metadata, building and exposing queries (prepared or parameterizable) as well as presenting query results	F, R
Front-end technologies	Libraries and frameworks for developing the user-facing interfaces of the platform, e.g. for creating user interfaces for visualisation, acquisition, downloading and data transfer, etc.	F, A, I, R

The Fair4Fusion demonstrators build on the technical requirements of MS8: Preliminary Requirements Document, while focusing on different aspects of a future system and evaluating different technologies. Furthermore, a summary of the technical requirements of the project is provided as an appendix, below. The appended is an excerpt of an online document which is being refined continuously. The demonstrators (WP5), to the degree that is possible due to their nature and to external non-functional conditions (e.g. available resources, central decisions needed, etc.), will primarily respond to at least these requirements designated by “MUST”. They may also cover additional functionality to promote the goals of the project as a Support Action.

Work performed in the framework of WP3 was the collection of requirements regarding a future Fair4Fusion system. Initially, the domain experts defined their expectations for the system in a form of high-level user stories, and then this information was iteratively fine-tuned and reshaped into a form of specific technological requirements (Appendix I), as also met by MS8. The methodology and technologies are compatible with European initiatives such as the European Open Science Cloud but they are also tailored to EUROfusion as an EIROforum[3] member.

[3] https://www.eiroforum.org

Fair4Fusion works in line with RDA/FAIR principles[4], similar to research infrastructures such as ICOS[5] and their Data Portal[6].

This section introduces and organises potentially relevant technologies to the Fair4Fusion demonstrators, and therefore also to the blueprint architecture, for each technology category listed in Table 1. Table 2 presents the main technologies that ought to be foreseen in the Fair4Fusion architecture.

3.1 Authentication and Authorisation technologies

The Authentication and Authorisation mechanism that ought to be integrated in the platform should support open, secure standards, allow a relative flexibility in selecting single-sign-on and institutional access permissions and be compatible with important third-party platforms, such as EOSC. Major protocols that could be supported include OpenID, OAuth2.0 and SAML2. In this technology category, promising implementations appear to be KeyCloak[7] as a means for IdP[8], eduTeams (using Perun[9]), EGI CheckIn, Indigo IAM, B2Access (based on Unity IDM[10]) as the means for AAI Community Proxy service.

3.2 Data representation and annotation

To facilitate and promote FAIRness it is critical to adopt standardised and formal terminologies for data description. Furthermore, the data itself should be stored or transformed to open formats while being readable by both machines and humans. Regarding metadata descriptions, we distinguish between the following types of metadata:

- **General information**: It refers to high-level annotations for the authorship, ownership, scope and history of the described resource

Semantic Resource	Scope
Dublin Core Terms[11]	A vocabulary for describing digital and physical resources
FOAF[12]	An ontology for the description of persons, their activities and their relations

[4] https://www.rd-alliance.org/fair
[5] https://www.icos-cp.eu
[6] https://data.icos-cp.eu
[7] https://www.keycloak.org/
[8] https://www.keycloak.org/
[9] https://perun-aai.org
[10] https://www.unity-idm.eu
[11] https://dublincore.org/specifications/dublin-core/dcmi-terms/
[12] http://xmlns.com/foaf/spec/

PROV-O[13]	An OWL (Web Ontology Language) representation of the PROV data model for expressing and exchanging provenance information on the web

- **Dataset characteristics**: This category deals with characteristics of the resource from the dataset perspective, e.g., its size, formats, information included, usage, licensing etc.

Semantic Resource	Scope
DCAT[14]	An RDF vocabulary for describing and sharing information regarding the structure and content of data catalogues
schema.org[15]	Schemas for structured data markup towards producing metadata connected to web resources

- **Domain-specific information**: This metadata category entails semantic resources specialised in the domain and its sub-domains, e.g. the ITER Physics Data Model.

Semantic Resource	Scope
ITER Physics Data Model (PDM)	The Data Model is used to describe ITER experimental and simulated data, as well as being able to describe data from any other fusion experiment. The data structures are identical for experimental and simulated data facilitating the simulation/experiment comparison.
Summary IDS	The PDM defines a number of *Interface Data Structures* (IDSs), providing access to different types of data and metadata. The *Summary* IDS in particular is of significance to Fair4Fusion as it encapsulates largely shareable information about experiments which can be used to make access to Fusion data FAIRer.

[13] https://www.w3.org/TR/prov-o/

[14] https://www.w3.org/TR/vocab-dcat-2/

[15] https://schema.org

3.3 Persistent storage mechanisms

Fair4Fusion demonstrators are designed around the Summary IDS schema and are therefore required to store summary metadata for the purposes of FAIRness, with an emphasis on findability. To this end, the following categories may be considered:

a. NoSQL databases use storage and retrieval mechanisms different from traditional SQL management systems. In general, they are better suited for distributed, clustered storage environments while providing a relative flexibility in incorporating data collections that follow different conceptual designs. A prominent solution supporting NoSQL is MongoDB[16], while Apache Cassandra[17] is a powerful open source alternative

b. Graph databases: A distinct paradigm of NoSQL is graph databases that organize data in graph structures, i.e. nodes and directional edges between these nodes. Graph databases are appropriate for storing semantically rich data such as RDF triples where the respective database management systems are called triple stores. Additional formalisms - like quadruples - are supported by a number of tools. Virtuoso[18] is one of the most mature triple store systems that accommodate the management of different database types. GraphDB[19] is another triple store solution based on RDF4J libraries that offers good standard conformance and full support to the SPARQL query specification. Neo4J[20] is also an increasingly popular graph database that follows a different paradigm for graph representation while being compatible with W3C specifications and thus, is a viable alternative as an RDF store.

c. Relational databases: Traditional database management systems are quite popular offering a plethora of tools for optimising, indexing and querying. Since their adaptability to semantic representation is inhibited by their tabular nature, they are not easy-to-use for storing semantically rich data and metadata. Nevertheless, their ubiquity and extensive support render them a viable candidate for the storage layer of the Fair4Fusion proposed platform. Well-known, free and widely used RDBMs are MySQL[21] and PostgreSQL[22].

3.4 Search and visualisation

Having established the representation and storage technologies to be used, a critical functional requirement is the presence of a stable and efficient search indexing technology on top of the accessible repositories, as well as a framework for presenting the retrieved results using different facets and modalities (text, diagrammatic, aggregates).

[16] https://www.mongodb.com

[17] https://cassandra.apache.org

[18] https://virtuoso.openlinksw.com

[19] http://graphdb.ontotext.com

[20] https://neo4j.com

[21] https://www.mysql.com

[22] https://www.postgresql.org

3.4.1 Search indexing frameworks

The pre-eminent library for building search engine platforms over large-scale, possibly heterogeneous, data collections is Lucene[23]. It implements high-performance indexing functions, as well as, advanced search algorithms for ranked, fielded, faceted and fuzzy searching.

Lucene is the foundation of multiple, full-fledged search server solutions, like Solr[24] and ElasticSearch[25], which provide complete and optimised control of search applications, including APIs for using the search stack. ElasticSearch, in particular, has also been successfully used as part of the JET dashboard, an internal UKAEA system used as the basis for the Fair4Fusion demonstrators' UIs. In addition to specialised indexing systems and frameworks, traditional database management systems, such as PostgreSQL and MySQL, increasingly support full-text searching.

3.4.2 Data visualisation frameworks

Data visualisation libraries and packages exist for every major programming language; thus, the development of a custom solution is a viable approach. However, complete data search, retrieval and visualisation platforms like Kibana[26], Grafana[27], Tableau[28], Splunk[29] and Cyclotron[30] appear to be promising in terms of presentation and close integration with different search engine frameworks. In addition, programmatic libraries, such as matplotlib[31], seaborn[32] and bokeh[33] are also very flexible and can be integrated with different Web and desktop technologies. The programmatic libraries in particular are being used as part of the demonstrators with demonstrator 2 focusing on evaluating different approaches.

3.5 Front-end frameworks

The development of the user-facing toolkit of the proposed Fair4Fusion platform needs, on one hand, the integration of a site-building environment and, on the other hand, the examination of web development libraries and frameworks to create custom applications and interfaces relevant to the user requirements.

[23] https://lucene.apache.org
[24] https://lucene.apache.org/solr/
[25] https://github.com/elastic/elasticsearch
[26] https://www.elastic.co/kibana
[27] https://grafana.com
[28] https://www.tableau.com
[29] https://www.splunk.com
[30] https://www.cyclotron.io
[31] https://matplotlib.org/
[32] https://seaborn.pydata.org/
[33] https://bokeh.org/

Regarding site-building environments, a potential candidate offering seamless integration with code repositories is Hugo[34]. Other, widely used solutions, like WordPress[35], Joomla[36] and Drupal[37] are worth mentioning as they offer significant resources for the platform presentation layer though they appear to be less flexible in incorporating custom modules which is an important requirement in the proposed implementation.

As far as it concerns the web development framework, there is a wealth of JavaScript libraries - such as node.js[38] and jQuery[39] - for implementing custom web modules and applications as well as similar packages for other programming languages like Django[40] for Python, JavaEE[41] assets for Java, etc. All these frameworks provide sufficient functionality for developing the required Fair4Fusion interfaces where the final selection will depend upon the developer familiarity as well as the existence of appropriate bridges/APIs with the underlying search framework.

3.6 Summary of 3rd-party technologies under consideration

In the following table, the technical solutions and frameworks considered for the implementation of the Fair4Fusion platform are tabulated. These are under active investigation, especially with regards to the blueprint architecture to be proposed in D3.2 *Blueprint Architecture for a Fusion Open Data Framework*.

Table 2: Summary of candidate technologies per technology class

Technology	**Solutions**
AAI	Eurofusion AAI, KeyCloak, eduTeam alternative technologies: EGI CheckIn, B2Access, Unity IDM, Perun, and others
Data representation and annotation	Dublin Core Terms, FOAF, PROV-O, DCAT, schema.org
Persistent storage	MongoDB, Cassandra, GraphDB, Virtuoso, Neo4J, MySQL, PostgreSQL

[34] https://gohugo.io
[35] https://wordpress.org
[36] https://www.joomla.org
[37] https://www.drupal.org
[38] https://nodejs.org/en/
[39] https://jquery.com/
[40] https://www.djangoproject.com/
[41] https://www.oracle.com/java/technologies/java-ee-glance.html

Search and visualisation	Lucene, Solr, ElasticSearch, Kibana, Grafana, Tableau, Splunk, Cyclotron
Front-end technologies	Hugo, WordPress, node.js, jQuery, Django, JavaEE

4 Existing Assets

4.1 Data Collections and their technical characteristics

Data generated by fusion experiments are provided in a variety of formats, with some being site-specific (e.g. IPX files on MAST). For the purposes of the Fair4Fusion demonstrators we have been making use of data originating from two different experimental facilities: provisionally MAST and WEST. The format characteristics, as they also appear in D1.6 *Data Management Plan,* are as follows:

Site	Format	Description
MAST	IPX	Used for storing video information - essentially a series of uncompressed JPEG200 frames with timing metadata
	netCDF/HD5	Self describing structured file
	IDA3	Locally defined file format
WEST	IMAS	ITER style hierarchical data structures
TCV	MDSPlus	Widespread file-based hierarchical data structure in the fusion community
AUG	CSV	Locally defined CSV schema for metadata, to help with rapid prototyping

4.2 Demonstrator environments and tools

One vehicle for Fair4Fusion to communicate concepts related to Open Science, and shape its future for the EUROfusion community, is via the delivery of two demonstrator systems which are currently under implementation in the framework of WP5. The two demonstrator systems are also useful as tools for trying out different technologies and practices responding to requirements. The Fair4Fusion demonstrators are documented as part of reaching MS15.

4.3 Semantic annotations

When using semantic technologies to their full potential, the data is modelled using ontologies that are interlinked among them. However, in many cases this is not possible because the data already exists and is stored in database systems or it is provided by legacy systems that cannot be modified and which use internal models (or schemas) to represent it. Nevertheless, there are methods and tools that can a) link existing structured or unstructured information to specific ontologies or b) transform/map the data to a semantic representation. This section is focused in the former.

Semantic enrichment, also known as semantic annotation (or tagging), enhances the source data with a context that is linked to some structured knowledge of a domain or application (ontology), which can be then exploited by different applications and services. This is done by attaching additional information to various concepts (e.g., people, things, places, organizations, etc.) in a given text or any other content. Since the newly discovered knowledge is described by standard ontologies, stored in machine-readable format and accessible through standard APIs and protocols, it can also be used for further machine processing allowing better integration with existing knowledge bases and their publication in the Linked Open Data (LOD) Cloud, discovering and understanding relations and dependencies between resources, as well as the implementation of all types of user scenarios. The process of semantically enriching data enables, thus, not only content reuse but also the inference of new knowledge. For instance, it can support the matching discovery between data elements, overcoming the differences among different structures and providing a solution for the (semi) automatic information integration and systems interoperability[42].

There are several methods and tools for semantic annotation that include a few – listed below – enabling the linking of existing structured or unstructured information to specific ontologies and may be relevant to Fair4Fusion, those are:

- JSON-LD[43] provides a way of linking structured information in JSON format to specific concepts in an ontology. It is a lightweight Linked Data format, easy for humans to read and write. By adding semantic annotations to JSON documents in a way that preserves their original structure, it provides a way to help JSON data interoperate at Web-scale. However, JSON-LD is just a method for encoding the linked data, not a tool to generate or discover the links between JSON elements and an ontology.
- DBpedia Spotlight[44] is a tool for automatically annotating mentions of DBpedia resources (i.e., almost all concepts from the domain of general knowledge, as well as

[42] https://www.ontotext.com/knowledgehub/fundamentals/semantic-annotation/
[43] https://json-ld.org/
[44] https://www.dbpedia-spotlight.org/

some concepts from specific domains) in text by performing named entity extraction, including entity detection and name resolution. It provides a solution for linking unstructured information sources to the Linked Open Data Cloud through DBpedia, as DBpedia is a hub of the LOD Cloud having links from and to many other datasets.

- GATE (general architecture for text engineering) is another tool for semantic enrichment. GATE[45] is an open software consisting of a family of tools for text processing. It has a set of reusable processing resources for common natural language processing (NLP) tasks, including the Information Extraction system (ANNIE)[46] enabling semantic enrichment of textual content.
- COGITO[47] is a commercial solution enabling semantic enrichment of content leveraging pre-trained vertical models and out-of-the-box and customizable taxonomies. Cogito is built on a knowledge graph (Sensigrafo)[48], where concepts (syncons) are represented as groups of lemmas with the same meaning. Syncons are interconnected through semantic and linguistic relations like hypernymy, hyponymy and other properties. Among other purposes, Cogito leverages the knowledge contained in Sensigrafo to disambiguate the meaning of a word by recognizing its context.

- Babelfy[49] is another tool which provides a unified, multilingual, graph-based approach to Entity Linking and Word Sense Disambiguation based on a loose identification of candidate meanings coupled with a densest subgraph heuristic which selects high-coherence semantic interpretations. Babelfy is based on the BabelNet multilingual semantic network [3] and jointly performs disambiguation and entity linking. BabelNet[50] is both a multilingual encyclopedic dictionary, with lexicographic and encyclopedic coverage of terms, and a semantic network which connects concepts and named entities in a very large network of semantic relations, made up of about 16 million entries, called Babel synsets. Each Babel synset represents a given meaning and contains all the synonyms which express that meaning in a range of different languages. Babelnet covers 284 languages and is obtained from the automatic integration of several sources including Wordnet, Wikipedia, Geonames, etc.

As a general rule for the above-mentioned approaches, the first step is the identification or definition of the ontologies that are to be used. From the above, JSON-LD is a domain independent way of embedding RDF-style semantics into JSON (depending on the context, different ontologies will be used). DBpedia Spotlight relies on the DBpedia knowledge graph (semantic version of Wikipedia), COGITO relies on their closed Sensigrafo knowledge graph, in GATE users can specify the underlying ontology, and Babelfy is based on the BabelNet

[45] https://gate.ac.uk/
[46] http://services.gate.ac.uk/annie/
[47] https://expertsystem.com/products/text-analytics-software-cogito-discover/
[48] http://expertsystemtraining.com/
[49] http://babelfy.org/
[50] https://babelnet.org/

multilingual semantic network made up of about 15 million entries from sources like Wikipedia, Wordnet, Wikidata and others.

5 Recommendations and proposed action plan

Taking into account the requirements produced in D2.3, the document accompanying MS8, Appendix I, as well as the identification of existing technologies and frameworks, we propose the following action plan for moving forward with two distinct yet adjacent technical objectives of the project:

- the implementation of the technical requirements that will be met by the Fair4Fusion demonstrators;
- by extension, the process for evaluating and selecting the specific technical solutions that will be incorporated in the blueprint architecture to be proposed.

5.1 Directions of the Fair4Fusion demonstrators

The Fair4Fusion demonstrators are being developed to respond to different contexts and build on different sets of assumptions. They also have common aspects, e.g. their UI is inspired by the JET dashboard. Both demonstrators aim to cover the same overall requirements, including data representation and modelling, visualisation, etc, however for experimentation and evaluation purposes they prioritise complementary technological aspects.

Currently, two demonstrators are being built, both containerised and readily deployable in arbitrary cloud resources:

1. One is built on IMAS technologies available on the EUROfusion Gateway[51]: the EUROfusion Gateway is a technological hub and environment for EUROfusion users. It comes with mature as well as developing technologies. Users must be registered on the Gateway and some of the underpinning technologies may be subject to stricter policies regarding maintenance, updates, etc.
    a. UDA – service for exposing the experimental data from experiment sites
    b. CatalogueQT: a DB directly supporting IDS summaries, and being responsible for harvesting the data
    c. Python-Django and RESTful APIs
2. One built independently of the EUROfusion Gateway. In this case, some of the core technologies may need to be developed, however they will be tailored to the requirements of the demonstrator and be less affected by other parallel developments in the EUROfusion Gateway. On the other hand, this way there is more room for experimentation and for trying out novel approaches.
    a. Minimal IDS-inspired DB
    b. Python-Django backend and RESTful APIs

[51] https://portal.eufus.eu

c. Visualisation and searching

5.2 Technology selection process for finalizing the Fair4Fusion blueprint architecture

Taking into account the maturity of the already available technical assets discussed in Section 4, as well as the lessons learnt from the development and evaluation of the Fair4Fusion demonstrators, the process for selecting technologies to be proposed for the blueprint architecture entails the following steps:

1. Identify critical technology classes where an existing solution is not readily available or mature enough
2. Organise small-scale test beds for assessing the functional and performance adequacy of each candidate technology in this class, taking into account results from the development and usage of the demonstrators
3. Review non-functional aspects of the aforementioned candidate technologies, e.g. availability and support, connectivity as well as ease-of-use
4. Select the best technology to be incorporated in the blueprint architecture

6 Conclusions

This report provides a survey of technologies and technological directions in fulfillment of the requirements and user stories identified by WP2. These technological directions drive the development of the Fair4Fusion demonstrators, based on technical requirements elicited in WP2. The demonstrators (WP5) primarily respond to at least these requirements designated by “MUST” – see Appendix. They may also cover additional functionality to promote the goals of the project as a Support Action.

This report as well as linked activities, e.g. the implementation of the Fair4Fusion demonstrators, inform the interim report on the blueprint architecture (MS10), the Blueprint Architecture for a Fusion Open Data Framework with proposition of the implementation plan (D3.2), and, crucially, future developments towards making Fusion research FAIRer.

Appendix I: Technical requirements summary (Ongoing effort – live document)

Index: Use Case number as it is listed in Deliverable D2.2 – ‘Interim Report on Open Science Use Cases for Fusion Information’

Type: Functional / Non-functional

Category: Interface / Management

Importance: Priority in MUST: 1-5

Priority in SHOULD: 1-5

Priority in MAY: 1-5

Index Naming: use case number.use case description line (https://docs.google.com/document/d/1RFmx41WeXfDnd6MM_GJ4IIGPm2OTIEaZl4I3GURGYhl/edit). requirement number for this one line

Importance: MUST: Priority inside MUST 1-5, SHOULD: Priority inside SHOULD 1-5, MAY: Priority inside MAY 1-5

Type: Functional/Non-functional

Category: Interface/Management

Index	Type	Category	Importance	Priority	Description
1.1.1; 2.5.1	Functional	Interface	MUST	1	Reporting functionality based on the provided criteria like shots per day or per year that are performed by each of the experiments. Exposed as rest API or the web representation using rest API as backend. Parametrizable selection of output information. As this potentially leads to very complex interfaces, a selection of fields to be included as choices maybe should take place before building the functionality.
1.2.1; 1.5.1	Functional	Interface	MUST	1	Search functionality for the particular parameters of discharges. The functionality should allow to choose limited set of global parameters as search criterias (e.g. global quantities). Multi-search.
1.2.2	Functional	Management	MUST	1	The system must allow the management (store, retrieve, modify) of metadata for different entities, i. e., discharges, publications, devices, scientitsts, etc. This functionality must be exposed via some API, and also via the user interfaces (Web Interfaces)
1.3.1	Functional	Interface	SHOULD	3	Search functionality of the publications related to each shot
1.3.3	Functional	Interface	SHOULD	3	Web interface for updating the metadata related to publication
1.3.4	Functional	Interface	SHOULD	3	Reporting functionality presenting the number of publications per shot, etc. Exposed as rest API or the web representation using rest API as backend.

1.3.5;
1.8.3;
4.1.2; 4.2.2; 4.3
2.5.2;
5.1.3; 5.2.3; 5.3
5.4.2;
6.1.2; 6.2.2; 6.3
7.10.2
	Functional	Management	SHOULD	1	The system should allow capturing and managing associations between entities, such as those related to a discharge like publications, devices, scientists, etc. in a single information unit representing the associated experiment that can be shared/cited/downloaded. This functionality must be exposed via some API, and also via a Web interface

1.4.1; 3.5.1; 3.6.1; 3.7	Functional	Interface	MAY	1	compare functionality of the discharges of the various devices compare with respect to key figures of merit. First the ability of the search of the discharges to be compared basing on some criteria
1.5.2	Functional	Interface	MAY	2	The system should display the cost of execution for each discharge, and when requested also detailed information including the cost breakdown
1.6.1	Functional	Management	MAY	2	The system should provide the functionality (via API & Web Interface) to assess the success of the discharge in terms of the fullfilment of its expectations against a set of conditions (metric/criteria)
1.6.2	Functional	Management	MAY	2	The system should support the specification of (multiple) success metrics/criteria to be used to assess a discharge success with respect to the fullfilment of its expectation. Each discharge may have different criteria, which may be specefied via an API or model

1.7.1	Functional	Management	MAY	2	The system should provide the functionality to assess the reliability of a device in terms of, e.g., how many (un-)successful shots it produced or on the success/unsucces ratio.
1.7.2	Functional	Interface	MAY	5	The system may display the reliability of a device throughout time in the Web interface
1.8.1	Functional	Interface	MAY	5	The system may provide the reporting of disruptions for each device, including the number of discharges that ended up in a disruption (in the device)
1.8.2	Functional	Interface	MAY	5	The reporting of disruptions for each device, should enable to display detailed information when requested, e.g., details of discharges leading to disruption, and to compare the discharges leading to the disruption to identify trends/patterns
4.1.1; 4.2.1; 4.3 3.1.1; 5.1.1; 5.2.1; 5.3	Functional	Interface	MUST		Be able to select and plot a given type of profile data (typically 1D radial + 1D time), from data entries previously selected from a search on the metadata. These data entries may correspond to different pulses of a given or of different experiments, allowing for graphical comparison of the selected profile quantity. Plot equilibrium information. Plot description of current profiles for particular shot at particular time. Plot the heating and current drive sources as a function of space and time for particular shot at particular time
4.1.2; 4.2.2; 4.3 3.1.1	Functional	Interface	SHOULD		Be able to choose different representations of profile data: 2D, slice along time, slice along radial position
4.5.1; 3.2.1; 3.3.1; 3.4 3.8.2; 3.9.1; 5.1.2; 5.2.2; 5.3	Functional	Interface	MUST		Be able to download the plotted data in various formats
4.4.1	Functional	Interface	SHOULD		Be able to compare the different data related to the density profile for a particular machine (to compare 2 measurements). On most devices there are multiple diagnostics all supplying data about the density profile.
2.1.1	Functional	Interface	MUST	1	Search functionality for free text constructs, where partial matches are valid. The querying mechanism can also foresee approximate matches (case insensivitiy, hyphenation removal, etc.)
2.1.2	Functional	Management	SHOULD	2	The system should support (semi) automtic metadata enrichment, including capability to carry out natural language processing against the research object payload (i.e., artifacts related to the experimentation) to extract metadata (such as keywords) from human-generated content
2.2.1	Functional	Interface	MUST	1	search interface must provide the fiels required for declaring time spans. Presets can be useful for speeding up commonly requested timespans
2.2.2	Functional	Management	MUST	1	The metadata describing the shots must include information about execution time, associated device, flat-top phase duration
2.3.1; 2.4.1	Functional	Interface	MAY	2	Mechanism for declaring complex, parameterisable, aggregate queries. This could be reduced to a set of query templates where just the parameters are requested by the end-user, as it seems quite open-ended
2.6.1	Functional	Management	SHOULD	2	Query history and retrieval linked to each user profile.

3.8.1	Functional	Interface	SHOULD	2	Provision of annotation module and respective interface elements
3.8.3	Functional	Management	SHOULD	2	The system should allow capturing and managing annotations related to plots, as well as relations between plots and their corresponding shots. This functionality must be exposed via some API, and also via a Web interface
5.4.1	Functional	Interface	SHOULD	2	When searching for particular metadata, the system should allow to obtain specified parameters from the entire data tree (not only from metadata). The interface should allow to define these parameters. The result can be large and thus should be stored in adequante formats (e.g. HDF5)
5.5.1	Functional	Interface	SHOULD	2	A user should be able to register for updates of data regarding particular shot.
5.5.2	Functional	Management	SHOULD	2	System should send notifications to registered users whenever interested data elements change
5.5.3	Functional	Management	SHOULD	2	The system should keep track of changes in the shot with sufficient descriptive and provenance metadata.
5.6.1; 5.6.2	Functional	Management	SHOULD	2	System should provide information about previous versions of data, if they exist. The system should keep track of the evolution of resources like datasets, including their versions with provenance information and relations with associated resources (e.g., entities that used/produced it)
6.1.1	Functional	Interface	SHOULD	2	System should allow to add/remove/modify/display metadata for shots. Focus on public atributes
6.1.3	Functional	Interface	SHOULD	2	System should allow to add/remove/modify/display metadata for shots. Focus on private atributes. It should be possible to distinguish between user private metadata and public metadata.
6.1.4	Functional	Management	SHOULD	2	The system should provide granular access control mechanisms at the level of resources (e.g., shots, data) and metadata elements, which may be specified via additional metadata
6.2.1	Functional	Interface	SHOULD	2	System should allow to add/remove/modify/display arbitrary public data for shots
6.2.3	Functional	Interface	SHOULD	2	System should allow to add/remove/modify/display arbitrary private data for shots
6.3.1	Functional	Management	SHOULD	2	System should allow to add/remove/modify/display references to publications as metadata for a shot or shots. The functionality may be exposed as REST API or as Web Interface
7.1-7.4.1	Functional	Management	SHOULD	2	Remote access to data stored on experimental sites or gateway should be provided.
7.5.1	Functional	Management	SHOULD	2	The data from experiments should be protected. Only data classified for save use in regards to law (e.g. GDPR) may be exposed by the system.
7.6.1	Functional	Management	SHOULD	2	The data creation/change at site/gateway should be properly propagated in the system
7.7.1	Functional	Management	SHOULD	2	The operations on data (e.g. movement) couldn't influence on plasma operations.

7.8.1	Non-Functional	Management	SHOULD	2	The hardware/manpower cost in institute related to the operation of the framework should be minimised. Additional cost related to opendata/F4F
7.9.1	Non-functional	Documentation	SHOULD	2	All data management solutions should be properly documented
7.9.2	Functional	Configuration	SHOULD	2	All data management solutions should be easly maintanable
7.9.3	Non-functional	Architecture	SHOULD	2	The data management procedures and solutions should be as future-proof as possible. Whenever it is allowed, the system should bese on widely used and well-supported components / standards.
7.10.1	Functional	Management	SHOULD	2	Embargo periods (on devices) should be enforced by the system
7.11.1	Functional	Management	SHOULD	2	The system should provide validation functionality for the data not yet out of embargo period to ensure that all paperwork has been completed
7.11.2	Functional	Management	SHOULD	2	The system should enable assessing the fulfillment of requiements of data artifacts embargo procedures
7.12.1	Functional	Management	SHOULD	2	The system should count the queries to specific data collections. In particular, the number of accesses to data of particular data providers should be counted
7.12.2	Functional	Interface	SHOULD	2	The system, via Web Interface, should allow to get information about the count of queries to specific data collections.
7.13.1	Functional	Management	SHOULD	2	The system should count the size of data accessed per specific data collections.
7.13.2	Functional	Interface	SHOULD	2	The system, via Web Interface, should allow to get information about the amount of data accessed per specific data collections.
8.1.1	Functional	Management	MUST	2	The system needs to ensure safe access to the data by application of Authentication and Authorisation procedures